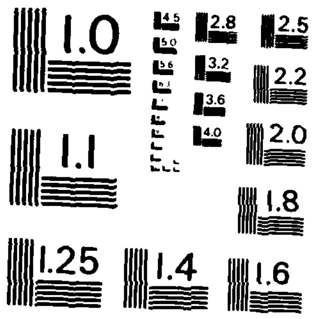

1.0
4.5
5.0
5.6
6.3
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

ARO 18690.5-EL

12

Center for Reliable Computing

Hard CPU Related Failures and System Activity: Measurement and Modelling

Ravishankar K. Iyer and David J. Rossetti

CRC Technical Report No. 83-6

(CSL TN No. 225)

May 1983

CENTER FOR RELIABLE COMPUTING
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305

This work was supported in part by the Department of the Army under Contract Number DAAG29-82-K-0105 by the Department of Energy under Contract Number DE-AC03-76F00515.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER CRC Tech. Rpt. 83-6	2. GOVT ACCESSION NO. AD-A130 821	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Hard CPU Related Failures and System Activity: Measurement and Modelling		5. TYPE OF REPORT & PERIOD COVERED Interim Tech. Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Ravishankar K. Iyer and David J. Rossetti		8. CONTRACT OR GRANT NUMBER(s) ARO DAAG-29-82-K-0105
9. PERFORMING ORGANIZATION NAME AND ADDRESS Center for Reliable Computing Computer Systems Laboratory Stanford University, Stanford, CA 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS DD Form 2222, Project No. P-18690-EL
11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office Post Office Box 12211 Research Triangle Park, NC 27709		12. REPORT DATE May 1983
		13. NUMBER OF PAGES 40
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Mr. James W. Gault Electronics Division U.S. Army Research Office P.O. Box 12211, Research Triangle Park, NC 27709		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

N/A

18. SUPPLEMENTARY NOTES

THE VIEW, OPINIONS, AND/OR FINDINGS CONTAINED IN THIS REPORT ARE THOSE OF THE AUTHOR(S) AND SHOULD NOT BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION, POLICY, OR DECISION, UNLESS SO DESIGNATED BY OTHER DOCUMENTATION.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Workload and Failure Measurement, Data Analysis, Statistical Models

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper describes the measurement and analysis of hard CPU and memory errors, and system activity at the Stanford Linear Accelerator Center computational facility. Nearly 25 percent of the errors were estimated to be permanent. The occurrence of a failure was found to be strongly correlated with the level and type of workload prior to the occurrence of the failure. For example, it is shown that the risk of a permanent error increases in a non-linear fashion with the amount of interactive processing. The observed tendency is present in three years of load data. This observation is significant because a (OVER)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) load-failure relationship found at the CPU level must, in our view, be considered fundamental. In addition, the fact that most of the errors are permanent, provides new information on these error types viz. their load dependent behavior. Our analysis procedure, used on the SLAC data, has been validated on an artificially created data base seeded with failures.

Hard CPU Related Failures and System Activity: Measurement and Modelling

Ravishankar K. Iyer and David J. Rossetti

CRC Technical Report No. 82-6

(CSL TN No. 225)

May 1983

CENTER FOR RELIABLE COMPUTING
Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California 94305

ABSTRACT

This paper describes the measurement and analysis of hard CPU and memory errors, and system activity at the Stanford Linear Accelerator Center computational facility. Nearly 25 percent of the errors were estimated to be permanent. The occurrence of a failure was found to be strongly correlated with the level and type of workload prior to the occurrence of the failure. For example, it is shown that the risk of a permanent error increases in a non-linear fashion with the amount of interactive processing. The observed tendency is present in three years of load data. This observation is significant because a load-failure relationship found at the CPU level must, in our view, be considered fundamental. In addition, the fact that most of the errors are permanent, provides new information on these error types viz. their load dependent behavior. Our analysis procedure, used on the SLAC data, has been validated on an artificially created data base seeded with failures.

Keywords: Workload and failure measurement, data analysis, statistical models.

CONTENTS

Abstract ii

page

1. INTRODUCTION 1

Related Research and Motivation 3

2. MEASUREMENTS 6

Error Measurement 6
Workload Measurement 8

3. OVERVIEW OF THE MEASUREMENT SYSTEM 10

Processing the Workload Data 13
Five minute intervals and smearing 13
Processing the Error Data (BUILD) 17
Combining Workload and Error Data (MATCH) 20
Summary of the Data Base 21

4. ANALYSIS 21

Workload and Error Analysis 21

5. THE LOAD HAZARD MODEL 26

6. HAZARD PLOTS 28

7. DISCUSSION AND CONCLUDING REMARKS 31

8. ACKNOWLEDGMENTS 33

REFERENCES 34

FIGURES

Figure page

1. Risk of error vs. system CPU usage (fraction). 3
2. CPU failures by hour of day (SLAC Triplex). 5
3. Detailed data flow in the measurement system. 12
4. Example of Smearing Algorithm 15
5. One Day of Batch CPU/Total CPU Data 17
6. Merging of Load and Failure Data 20
7. Frequency Distributions: System CPU (Fraction between 0,1) . . . 25
8. Frequency Distributions: IO (Per second) 25
9. Frequency Distributions: TOTCPU (Fraction between 0,1) 25
10. Hazard Plot: System CPU (Fraction between 0,1) 30
11. Hazard Plot: IO (Per second) 30
12. Hazard Plot: EXCP (Per second) 30

TABLES

Table — page

1. Sample error data (EREP) 8
2. Input data for workload variables. 10
3. Definitions of workload variables. 13
4. Clustering Statistics 19

1. INTRODUCTION

The highly interactive and diverse nature of modern day systems has made high reliability a central issue in computer system design. It is not, in general, feasible to guarantee a perfect system, either in hardware or in software. Accordingly, depending on the nature of the application, it is important to design into the system the ability either to continue operation in the event of a failure or to react to a failure in a predictable manner.

Theoretical models can only deal with a restricted class of problems. Most often it is the problems outside the range of theoretical models which cause the most severe malfunctions. Accordingly, at this stage there is no better substitute for results based on actual measurements and experimentation. An experimental study provides not only a view of the end product but also gives some insight into persistent problems. This information can be very valuable in designing new systems.

This paper describes the measurement and analysis of hard CPU and memory errors, and system activity at the Stanford Linear Accelerator Center (SLAC) computational facility. The authors' approach has been to start with a substantial body of empirical data on system load and failures. On the basis of these measurements several experiments were conducted to examine the dependence of hard failures on system activity. The salient features of the measurement process and important results are outlined below:

1. The present study concentrates on hard CPU related errors. A measurable number of the failures (between 15 - 25 percent)[1] were

[1] Between 75 - 85 percent of all errors were temporary (transient or intermittent) and are discussed in [Iyer 82b] and [Rossetti 81].

estimated to be hard failures (CPU and main memory).

2. The measurement process is automatic; it captures a detailed internal view of the system, especially under failure conditions.
3. From the measurements, a completely new data base of failures and workload was established. The workload and failure data were combined in order to match failures with workloads at the times of failure.
4. The measurements and statistical experiments clearly demonstrate a non-linear increase in the risk, of hard CPU related errors, due to increased values of workload variables. Examples are CPU utilization, input/output rate, and interrupt rates.

A representative measurement is illustrated in Fig. 1, which shows how an increase in the system CPU usage, SYSCPU, (a measure of the system overhead; a fraction between 0 and 1) can result in higher risk of hardware failures in the CPU and main memory. The horizontal axis is the workload variable; the vertical axis is the risk of error. Modeling details will be given later in this paper.

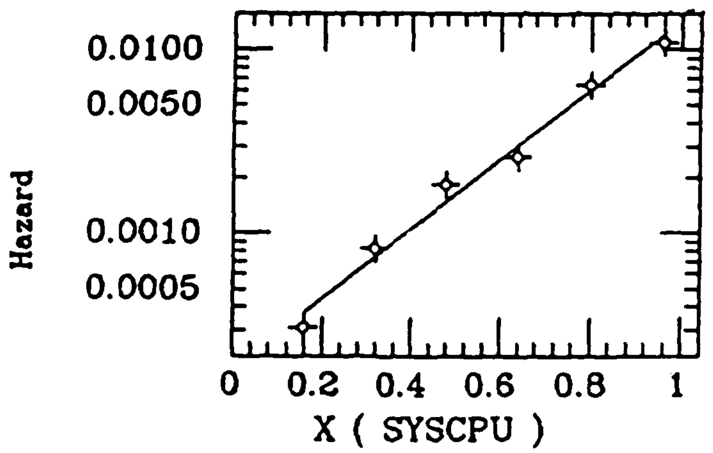


Figure 1: Risk of error vs. system CPU usage (fraction).

1.1 RELATED RESEARCH AND MOTIVATION

There is now considerable experimental evidence to show that computer reliability is a dynamic function of system activity (as measured by the workload). A number of studies [Butner 80], [Iyer 82a,b] and [Castillo 80, 81] provide statistical evidence on a number of machines to support this observation. Even though the exact nature of this dependency is not fully understood, it would appear that that computing systems, which need maximum reliability at their peak load, require a re-evaluation of their reliability projections.

An important, and as yet unanswered question is whether an increased level of system activity results in an increased level of hardware failures. In particular, it is important to determine whether hard failures in logic elements (CPU and storage) are also workload dependent i.e., does higher system activity result in a higher level of CPU and memory failures.

Some evidence to this effect was available from an early analysis of failures on the SLAC Triplex [Iyer 82a]. The study found a strong correlation between the occurrence of hard failures and the load on the system, as measured by variables such as the paging rate and the jobstep processing rate. All failures were considered, not simply the ones which led to system service interruptions. Most importantly the effects were such that the average failure rate for various system components varied cyclicly over a band of significant width as determined by the daily load variations. Fig. 2 below is a representative histogram, from that study, of all hard CPU failures plotted by the hour of day, averaged over 1978.

Subsequently a more detailed and accurate study was performed on all CPU errors [Iyer 82b]. It was found that all errors which affected the CPU correlated strongly with system activity. The large majority of these errors however, (75 - 85 percent) were temporary. More recent studies conducted on the IBM 3081 at SLAC found a similar behaviour with software related failures on VM/370 [Rossetti 82]. Additional substantiation of these results came from experimental studies on DEC systems reported in [Castillo 80].

There has been some effort at modelling the observed load/failure relationship. Possible cause-effect scenarios are discussed in [Butner 80] and [Iyer 82a]. Castillo and Siewiorek [Castillo 81,82] have proposed the use of a doubly-stochastic Poisson process to model the cyclic

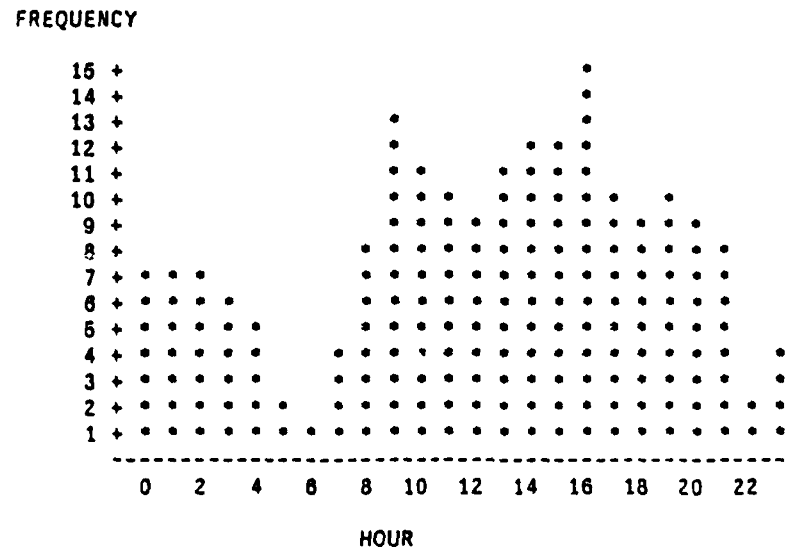


Figure 2: CPU failures by hour of day (SLAC Triplex).

load-failure relationship. The model assumes that the instantaneous failure rate can be described by a cyclostationary Gaussian process. In [Gunther 80] a novel theoretical model for an apparent dependency of failure on load, based on a random walk formulation, is described. There is no doubt that more detailed experimental results are necessary before a clear understanding of the observed behaviour is possible.

The next section discusses the failure and workload measurements taken and briefly presents the organization of the data. Subsequent sections describe the procedures employed to analyse hard failures and present new results. Finally, the paper summarises the important results and highlights the conclusions that can be drawn from them.

2. MEASUREMENTS

2.1 ERROR MEASUREMENT

As stated earlier, the present study uses the most detailed data from the log maintained by the operating system as errors are detected by the hardware and recorded by the software. High level system behavior, as seen by the computer operator and users, is not directly measured. Instead, there is much information on hardware errors, both permanent and non-permanent (transient and intermittent), as they occur in the detailed operation of system components.

The SLAC system, during the period of our study, consisted of two IBM 370/168 mainframes and an IBM 360/91 connected in a triplex mode. The data for our study, which consisted of three years of measurements (1979, 1980, and 1981), came from the two IBM 370/168 mainframes. The log referred to above is commonly called the "EREP" log, from the Environmental Recording Editing and Printing program used to accumulate and format it for maintenance [IBM 79].

Errors in IBM 370 systems are classified into three major types:

1. CPU Errors - In the central processor and storage.
2. Channel Errors - In I/O channels and associated interfaces.
3. Outboard Errors - In any device beyond the channel-control unit interface, i.e. all errors in I/O devices.

For each error, whether recoverable or not, the operating system creates a time-stamped record describing the error and providing relevant information on the state of the machine. As an example, for a CPU error, the state information might include the contents of all internal registers and diagnostic information collected by the hardware (such as parity indicators and error flags). At SLAC this information is collected on a daily basis and archived for many years.

Since the EREP log does not specifically provide information on hardware failures, it is necessary to estimate this information from the data. The following rule was used to estimate a hardware failure: If the machine check condition interrupted the CPU and re-occurred three times or more in sequence, the failure was considered to be a hardware failure in the CPU or main memory. The vast majority of these failures were storage errors. An examination with the repair log maintained showed good agreement. An important reason why this heuristic works is, due to the fact, that at high workloads a bad memory location is very likely to be rediscovered without much latency. In many cases it was found that this lead to a system termination. A sample of the hardware failure data obtained on this basis is shown in Table 1.

TABLE 1

Sample error data (EREP)

OBS	CPUMODEL	PROGID	MCCOND	STRGBITS	DMGLENGN	SYSSTAT	DMGAREA	ERRORTYP	UTIME	SYSDMG	IPDMG	SYSREC	TIMDMG	EXTDMG	DEGRD	STRGERR	RECY	DMG
305	168	ASPNUC	20	8F	0050	08	80	A0	17SEP79:08:26:07			RCVY				UNCORR	OSTER	RLSTRG
306	168	ASPNUC	40	8C	0050	08	80	A0	17SEP79:08:33:43		IDMG					UNCORR	OSTER	RLSTRG
307	168	ASPNUC	40	8C	0050	08	80	A0	17SEP79:08:36:47		IDMG					UNCORR	OSTER	RLSTRG
308	168	FORTRANH	20	0F	0050	80	08	00	28SEP79:19:42:33			RCVY					HWRCY	PROC
309	168	FORTRANH	20	0F	0050	80	08	00	28SEP79:19:43:01			RCVY					HWRCY	PROC
310	168	ASPVER	04	0F	0050	80	08	00	06OCT79:00:30:22					EDMG			HWRCY	PROC
311	168	RESOLVE	04	0F	0050	80	08	00	16NOV79:21:27:02					EDMG			HWRCY	PROC
312	168	FORTRANH	40	2C	0050	80	08	00	20NOV79:03:07:15		IDMG					KUNCOR	HWRCY	PROC
313	168	ASPNUC	40	2C	0050	80	08	00	20NOV79:03:07:15		IDMG					KUNCOR	HWRCY	PROC
314	168		40	2C	0050	08	00	84	20NOV79:03:07:15		IDMG					KUNCOR	OSTER	
315	168	MAIN	04	2F	0050	80	08	00	20NOV79:05:10:59					EDMG		KUNCOR	HWRCY	PROC
16	168	ASPNUC	20	2F	0050	80	08	00	20NOV79:05:10:59			RCVY				KUNCOR	HWRCY	FROC
317	168		40	2C	0050	08	00	84	20NOV79:05:10:59		IDMG					KUNCOR	OSTER	
318	168		04	2F	0050	08	00	84	20NOV79:05:10:59					EDMG		KUNCOR	OSTER	
319	168	MAIN	04	2F	0050	80	08	00	21NOV79:08:15:10					EDMG		KUNCOR	HWRCY	PROC
320	168	MAIN	20	2F	0050	80	08	00	21NOV79:08:35:03			RCVY				KUNCOR	HWRCY	PROC
321	168	MAIN	04	2F	0050	80	08	00	21NOV79:08:35:09					EDMG		KUNCOR	HWRCY	PROC

2.2 WORKLOAD MEASUREMENT

Since errors in processors occur fairly infrequently (on the order of once a day for our measurements), correlation with workload requires long term workload figures. The workload data comes from two sources: the built-in system utilization facility, and a software monitor written specifically for this study. They are discussed below.

The operating systems in the processors measured, use IBM's System Management Facilities (SMF) for usage accounting. SMF was originally designed to provide accounting information, but it has evolved over the years to include more general performance measurement information. SMF is discussed exhaustively elsewhere [IBM 73], [Butner 80] and will not be detailed here.

In general, SMF data consists of records giving resource utilization figures for jobs, files, I/O devices, and a potpourri of statistics gathered and written on a periodic basis. For this work we use the type 4 (Step) record, which holds statistics for each job step as it completes execution, and the type 1 (Wait) record, written roughly every 10 minutes, which summarises global system utilization during that 10 minute period. With careful processing SMF can provide excellent workload statistics, especially when high resolution results are not needed.

To obtain more detailed information about transient behavior in the CPU we implemented an interrupt rate monitor, called INTRACK. There are four classes of interrupts in the IBM 370 architecture:[2]

1. External (EXT) — Used by the operating system for clocks and inter-CPU communication.
2. Supervisor Call (SVC) — Caused by any SVC instruction. Used for operating system services, such as: memory allocation, synchronization, I/O, timing, etc.
3. Program (PROG) — Program traps due to arithmetic conditions (e.g. division by zero), invalid operations, or page faults.
4. Input/Output (I/O) — From completion of I/O operations.

The interrupt monitor (INTRACK) archived the interrupt data along with the SMF data described above. Table 2 summarizes the sources of data for the workload information.

[2] Machine check interrupts are not considered here because they are already collected in the EREP data.

TABLE 2

Input data for workload variables.

Record	When generated	Contents used
Step	At end of each batch job step	Accounting and job usage data, e.g. CPU time, No. of I/Os, memory usage.
Wait	Approx. every 10 minutes	CPU wait time during preceding 10 minute period.
INTRACK	Normally every 10 minutes (but settable)	Contents of four cumulative interrupt counters for: External, SVC, Program, I/O.

3. OVERVIEW OF THE MEASUREMENT SYSTEM

An objective of the measurement system was to make data management as automatic as possible so that it is unnecessary to know the particulars of operating systems, software monitors, record formats, and the like. The Statistical Analysis System (hereafter called SAS) [SAS 79] provided a rich environment for data handling, in addition to its procedures for statistical analysis. Once a few programs were written to capture and reduce the raw data, the information was immediately built into SAS data bases (called SAS data sets), on which the full power of SAS could be used to sort, select, merge, and extract information. More than 50 SAS programs, some very simple, were written to perform a variety of data handling operations on the data bases. This section discusses the system as a whole, describing the flow of data in general terms. Later, important components such as error clustering and workload smearing are covered in detail.

The transformation of raw workload and error data into usable data bases for analysis is performed by a collection of programs, some written in PL/I and many written in SAS. Refer to Figure 3 for the organization of these processors and the flow of data through them as they are described in the following sections.

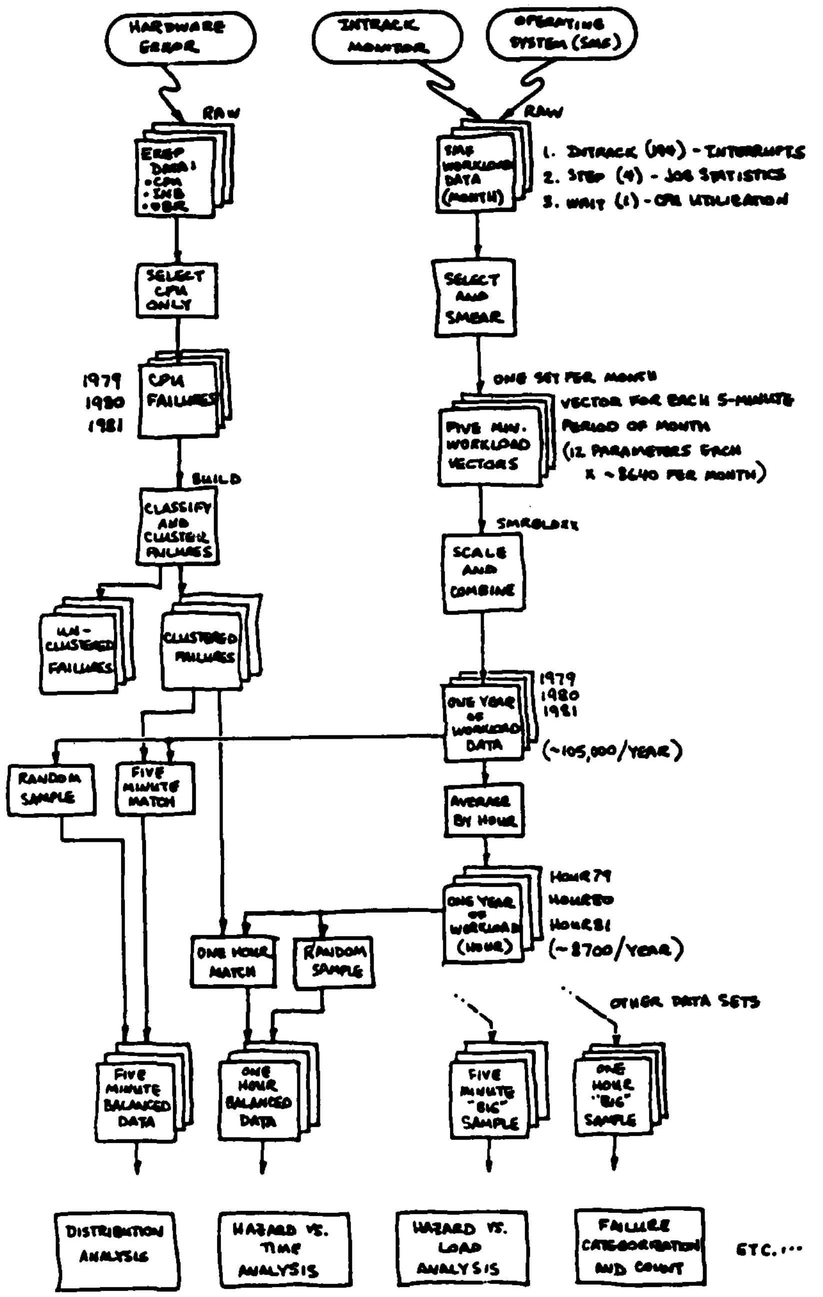


Figure 3: Detailed data flow in the measurement system.

3.1 PROCESSING THE WORKLOAD DATA

Workload processing begins with a program written to select and condense a specified set of SMF record types. This program is used to process the thirty reels of tape comprising the archived SMF data from 1979 to the present.

3.1.1 Five minute intervals and smearing

A number of workload variables are defined to provide estimates of various characteristics of system load throughout the three year measurement period. They are summarized in Table 3.

TABLE 3

Definitions of workload variables.

Name	Units	Derived From	Indicates
COREQ	KBytes	Summed Smear	Batch memory requests
COREU	KBytes	Summed Smear	Batch memory usage
VOLWAIT	sec.	Summed Prorated Smear	Batch I/O wait time
EXCP	1/sec	Summed Prorated Smear	Batch induced I/O load
PAGEI	1/sec	Summed Prorated Smear	Batch paging (in)
PAGEO	1/sec	Summed Prorated Smear	Batch paging (out)
BATCPU	fract.	Summed Prorated Smear	Batch CPU usage
SYSCPU	fract.	SMF Wait, BATCPU	Nonbatch CPU, Ovhd., etc.
TOTCPU	fract.	SMF Wait, Smeared	Overall CPU load
EXT	1/sec	INTRACK	Timer and clock activity
SVC	1/sec	INTRACK	Overall O.S. activity
PROG	1/sec	INTRACK	Paging/prog. exceptions
I/O	1/sec	INTRACK	Overall I/O activity

The workload time granularity was defined to be five minutes, meaning that for each five-minute period from January 1979 a vector of 13 workload variables was created. The process described below is applied to each of the variables. Essentially, the process takes what information is available in a record and distributes it into the time slots the record describes.

Each input record provides a starting time, an ending time, and a value for one or more load measures. Each of these measures is "smeared" into the five-minute bins defined by the starting and ending time of the event, either on a proportional basis (for variables representing counts or times), or directly (for "level" variables, such as memory usage). The algorithm also takes care of the subtle handling of partial bins at the interval endpoints, in addition to the case where both endpoints lie somewhere in the same bin. For these cases the amount accumulated into the bin is weighted by the fraction of time spent in the bin. Figure 4 presents an actual numerical example with four jobs overlapping in various ways. Notice that the height of each bin is the sum of the <u>time averaged</u> values of input values entering that bin. This averaging is similar to approximations that occur in numerical integration problems.

As stated earlier, the smearing is done one month at a time, with approximately 8640 bins per month, depending on the number of days in the month. Finally, the estimates are concatenated into one-year groups to form the "Five-Minute Smeared Data." For example, a complete day of smeared points (the 288 five-minute bins from Monday, January 5, 1981)

Example Smearing of the CPU Time of Four Jobs

Job	Start Time	End Time	Elapsed Time	CPU Time	CPU/Elapsed
A	0.5	4.7	4.2	2.0	0.48
B	3.3	9.5	6.2	7.0	1.13
C	4.1	5.8	1.7	1.2	0.71
D	7.1	7.8	0.7	0.3	0.43 (0.30)*

* Since job D is completely within a bin, its value is prorated into that bin's sum.

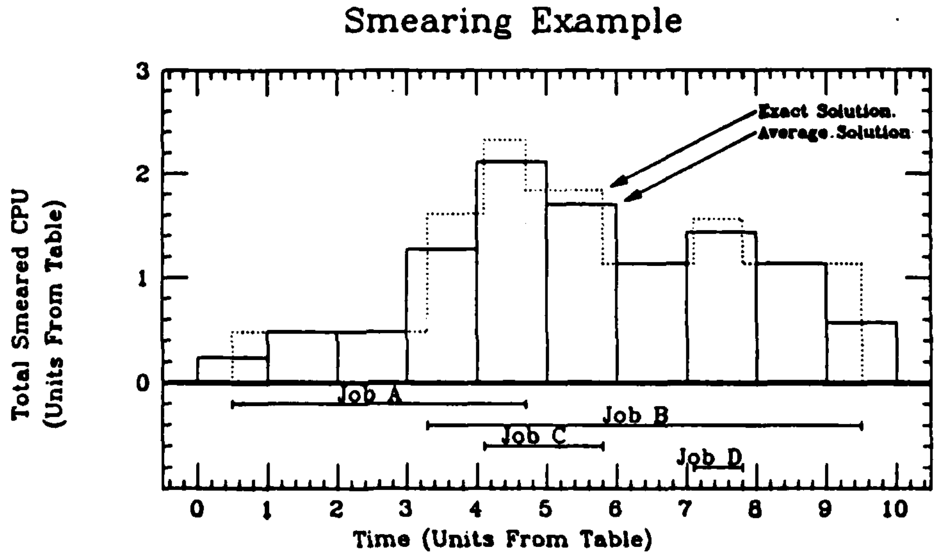


Figure 4: Example of Smearing Algorithm

for two variables is given in Figure 5. Each small step in the figure is a five-minute average; the solid upper line represents percent CPU busy, the dotted lower line is batch CPU. The plot shows the familiar early morning lull between 5 and 8 am with a dramatic climb to full utilization at about 10 am. Notice that in the evening, from about 10 o'clock on, batch work forms most of the CPU load, while during the day it is only in the 35 to 40% range with the remainder going to timesharing and overhead. It is also interesting to note that at a few rare points batch CPU seems to be greater than the total. This is due to the averaging algorithm's smearing of a job's CPU usage evenly over the job's duration while the total CPU figure is derived from a 10-minute global system total.

To study longer-range loading effects we also built a data base of one-hour smeared workload vectors. Each one-hour point is derived from the five-minute smeared data by averaging the twelve five-minute points in that hour and tagging the new point with the starting time of that hour. There are 8760 such vectors in a non-leap year. Another reason for creating the one-hour data is to test whether system crashes occurring soon after CPU failures cause the five-minute averages in the period preceding the failure to be artificially decreased. This could happen because jobs executing at the time of the crash would not contribute to the smeared totals as they should. A preliminary analysis showed this not to be a problem.

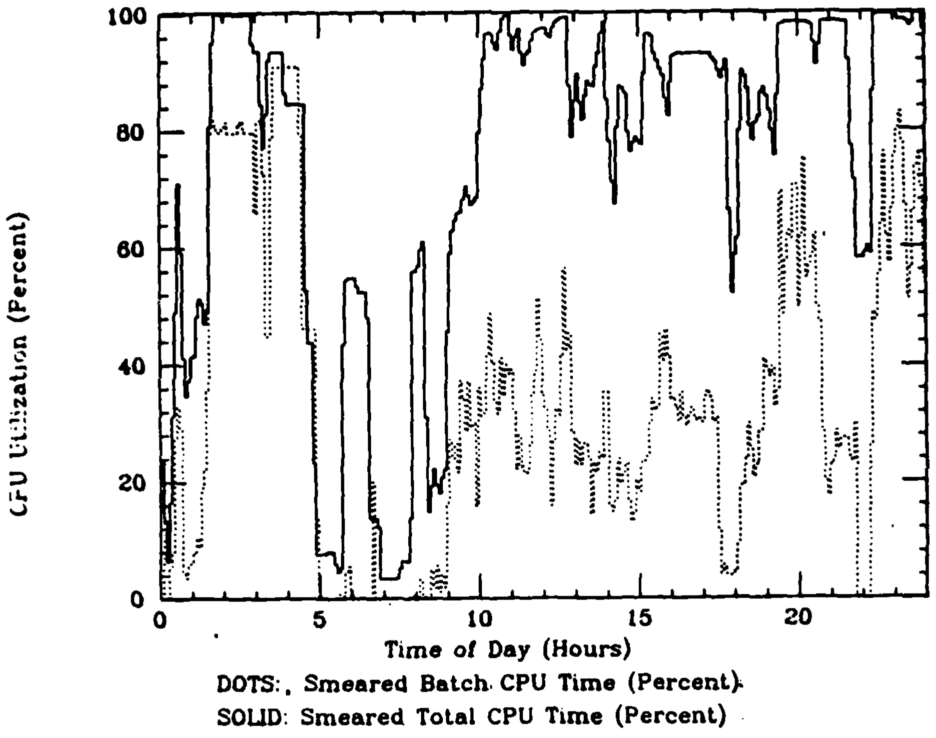


Figure 5: One Day of Batch CPU/Total CPU Data

3.2 PROCESSING THE ERROR DATA (BUILD)

This section presents the method used to process raw errors into the data base used for analysis. A SAS program, called BUILD, performs the following steps:

(i) Select: The raw EREP data includes CPU, channel, and device errors for all equipment in the installation. Only CPU (Machine Check) errors on the two 370/168s are selected for analysis.

(ii) Decode and Classify: In each MCH record there are a number of bits describing the type of failure, its severity, and the result of hardware and software attempts to recover from the problem. These bits are decoded into classes meaningful to this analysis and analyzed in later processing. General machine check status indicators are provided by the hardware are described fully in the System/370 Principles of Operation [IBM 81]).

(iii) <u>Sort By Processor and Time</u>: To facilitate clustering in the next step it is necessary to sort the data by CPU id (serial number) and time of failure within CPU id.

(iv) <u>Cluster</u>: Errors occurring within 5 minutes of each other were coalesced. For each error point, the following test was performed:

```
IF <error type> = <error type of previous error>
            AND <time away from previous error> $\leq$ 5 minutes
   THEN <fold this error into the cluster being built>
   ELSE <start a new cluster>.
```

The result is a set of clustered errors for each year. Associated with each cluster is information consisting of error classifications, number of points in the cluster, time of first and last errors in the cluster, and a variety of status data provided by the hardware and operating system.

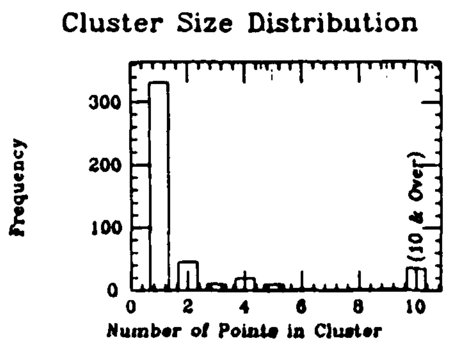
Some interesting things can be learned from a cursory analysis of the clusters derived as stated. Summary statistics for the number of points in a cluster (NPOINTS) and time spanned by a cluster (SPAN) are shown below in Table 4. Table 4 shows clearly that the clustering algorithm is having an effect by gathering long bursts of failures into a few large clusters, indicated by the maximum 192 points and 1310 second time span. The table also shows that lone failures predominate, with median cluster size of one and time span of zero, showing that the clustering algorithm is not artificially forcing them together. The accompanying bar charts also show this behavior. Clustering is important in the load-failure analysis to avoid biasing the results with repeated errors from the same failing component.

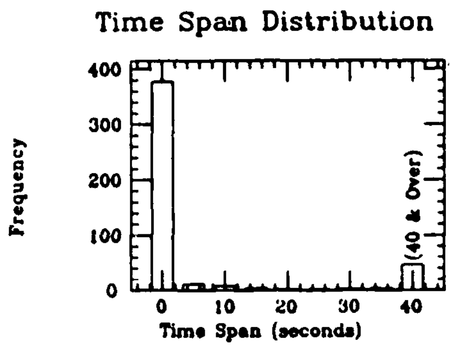
TABLE 4

Clustering Statistics

Original Errors: 1,903   Clustered Errors: 456 (76% reduction)

	NPOINTS	SPAN (seconds)
Mean	4.17	20.44
Median	1	0
90th Percentile	5	48.6
Minimum	1	0
Maximum	192	1310





3.3 COMBINING WORKLOAD AND ERROR DATA (MATCH)

The final and most important step of the data base building process is the matching of errors and workload. By matching we mean the combining of each error point with information on system workload at the time of the error. The clustered error points are processed sequentially and for each point: (1) The time of the five-minute interval preceding the error is calculated, and (2) used as a key to locate its corresponding workload observation. Then (3) the vector of workload variables from that observation is merged into the error observation.

In order to determine the load at the time of failure, the 5-minute load averages (which we refer to as smeared averages) were merged with the EREP log. The load at failure was taken to be the load in a five minute interval prior to the failure to eliminate perturbations from system error recovery or a system crash. The matching is shown in figure 6.

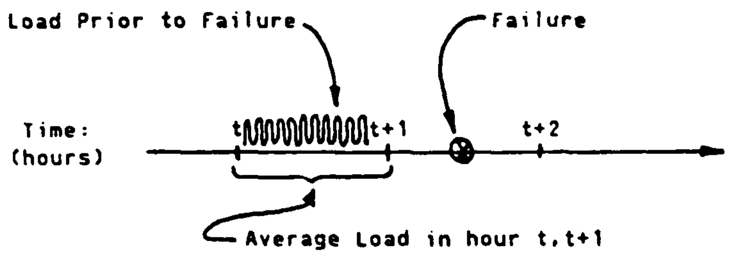


Figure 6: Matching failures and workload.

Note that the interval containing the failure is not used because of the measurement distortion that can be caused by error recovery activities, and the fact that the system may not continue to run after the error. Also, the exigencies of a system crash may prevent the operating system from gathering workload and accounting statistics.

In the case of one-hour averaged workload measurements, the algorithm is the same except that the previous hour's load is used.

3.4 SUMMARY OF THE DATA BASE

Summarising the above presentations, the following major sets of data were created:

- Clustered and unclustered "pure" errors - from which standard failure analysis can be drawn to obtain a number of statistics, e.g. mean time to failure, hazard with time, etc. See [Shooman 1968] for more information.
- Three years of workload information - also useful for studies not necessarily related to reliability. These points exist in both five-minute and one-hour granularities.
- Errors matched with workload - in both the five-minute and one-hour forms. These observations can be used to study the connection between load and errors in large computer systems.

4. ANALYSIS

4.1 WORKLOAD AND ERROR ANALYSIS

The data consisted of three years of load/failure measurements, 1979, 1980 and 1981. The 1981 data contains additional measurements made by our special purpose interrupt monitor. Initially, we analyzed each year separately. Since there was no significant difference in the 1979 and 1980 results, it was considered appropriate to combine the corresponding load-failure data. Of the thirteen workload measures collected for the study, four were chosen to be studied for 1979 and 1980. They were:

1. COREU — The sum of memory allocated by batch jobs (K bytes).
2. EXCP[3] — The I/O initiatation rate by batch jobs (I/Os per second).
3. SYSCPU — CPU utilization for system, i.e. non-batch, tasks (a fraction between 0 and 1).
4. TOTCPU — Total CPU usage (a fraction between 0 and 1).

For 1981 the following interrupt measurements were also included:

1. SVC — Supervisor calls (rate per second).
2. IO — I/O interrupts, completion of I/O operations (rate per second).
3. PROG — Program interrupts (rate per second).

Measures such as the SYSCPU and IO provide a measure of the system interactive load, while measures such as TOTCPU provide a general view of the CPU usage. The variable "BATCPU", derived from the difference between is a direct measure of batch usage.

Recall that the data base developed contains not only the values for the specified workload variables to a five minute resolution but also the values of the same variables matched with failure times. From this data two types of distributions were developed. The first, $\ell(x)$ is simply the distribution of the workload variable in question

$$\ell(x) = \Pr\{\text{workload} = x\}^{4}$$

[3] An acronym for "EXecute Channel Program"

[4] The workload (or load) is assumed to be a discrete random variable for this discussion.

The second is the joint distribution of failure and the workload measure:

$$f(x) = \Pr \{\text{failure occurs and load} = x\}.$$

In this expression, failures and load values are represented as they occur on an actual system, where favored loads contribute more to the distribution than loads of low probability. To remove this effect we divide f(x) by the associated load probability $\ell(x)$. Using the well known notion of a conditional probability distribution [Feller 68] we write

$$g(x) = \Pr \{\text{failure occurs} \mid \text{load} = x\} = \frac{f(x)}{\ell(x)}$$

Therefore g(x) can be thought of as the probability of a failure at a given load <u>when all loads are equally represented</u>; it is the conditional failure probability. In the figure g(x) represents the conditional probablities arranged by increasing x (workload). Note that, since each of these probablities are calculated independently, g(x) is not a probability distribution in the regular sense of the term.[5]

---

[5] A commonplace analogy to illustrate the above distinction is that automobiles travelling at 150 mph have a higher probability of accident than those travelling at 55 mph. However, there are far more accidents for autos going 55. To obtain an accurate representation of the risks involved in travelling at high speed, we must divide the number of accidents occurring at each speed by the number of autos travelling at that speed.

Figures 7 and 8 depict the ℓ, f, and g distributions of System CPU (SYSCPU) and I/O and Total CPU (TOTCPU) for 1981.

As a general observation we note that, where the difference between $\ell(x)$ and f(x) is considerable, we might expect to see a workload dependency in the failures. If $\ell(x)$ and f(x) are similar, the relationship is probably not significant. A g(x) distribution weighted in favor of higher workload values will clearly generate a higher risk of failure as the load increases.

It would appear from the g(x) plots for SYSCPU and IO that higher values of these measures (> 50 for IO) contribute more significantly to hard failures than the lower values. Examining the plots for TOTCPU we note that, as measured by CPU utilization, the system was heavily loaded most of the time. The $\ell(x)$ and g(x) plots for TOTCPU show considerable similarity. It would therefore appear from this cursory analysis that failures are not induced by higher execution rates, as measured by CPU usage alone.

In order to quantify this effect, in particular to determine exactly the risk or "hazard" associated with higher workload values, we employed what we refer to as a "load hazard" model, the development and application of which is discussed in the next section.

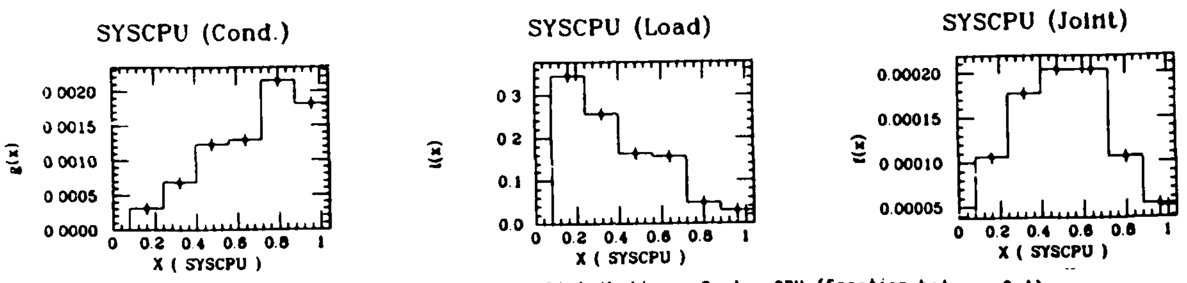


Figure 7: Frequency Distributions: System CPU (Fraction between 0,1)

Low loads are favored (ℓ(x)) while the joint distribution f(x) is almost symmetrical. The resulting conditional distribution g(x) shows a clearly increasing probability.

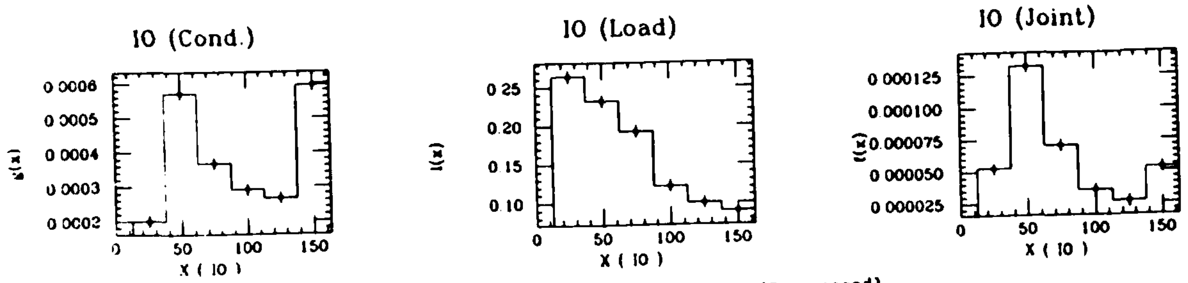


Figure 8: Frequency Distributions: IO (Per second)

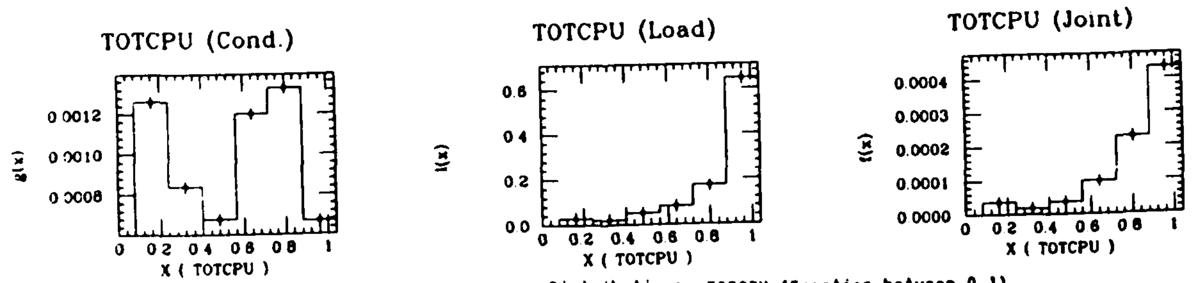


Figure 9: Frequency Distributions: TOTCPU (Fraction between 0,1)

5. THE LOAD HAZARD MODEL

The object of the analysis was to determine:

1. Does a higher level of system utilization result in a higher risk of failure than a lower level?
2. Is the relationship linear with the workload variables, or is there a nonlinear increasing effect?

In practical terms, if such an effect exists, it is expected that the load will act as a stress factor. For this purpose we developed and validated a load-hazard model which formed the basis for our tests. A detailed description of the development and validation of this model appears in [Iyer 82b]. Briefly, an inherent load hazard z(x) is defined as

$$z(x) = \frac{\Pr\{\text{Failure in load interval } (x, x+\Delta x)\}}{\Pr\{\text{No failure in load interval } (0, x)\}} \tag{1}$$

In close analogy with with the classical hazard rate in reliability theory [Shooman 68], z(x) measures the incremental risk involved in increasing the workload from x to $x+\Delta x$[6] (e.g. if the system is currently

[6] In applying the load hazard model to our data we made a simplifying assumption that the workload monotonically increases until failure occurs. This is a conservative assumption which was made primarily to simplify some cumbersome aspects of the data analysis. It has the additional advantage of allowing us to estimate a lower bound on the workload related risk (if any). This is due to the fact that under the assumption of a monotonically increasing workload, factors such as cycling (between low and high usage) and other random variations are ignored. It is well known that such stresses only serve to add to the hazard rate [Kujowski 78], [Arsenault 80]. Thus by neglecting them we underestimate the hazard being measured.

operating at 80 percent of full load, as measured by CPU usage, what is the increase in the risk of failure, if the load is increased to 90 percent?)

The numerator of $z(x)$ was determined from $g(x)$. The survival probability in the denominator (i.e. the probability of no failure in the load interval $(0,x)$) was for practical purposes found to be very close to the probability of reaching a given workload or higher (determined from the workload distribution $l(x)$). This is simply due to the fact that, in our data, failure events are much fewer than the five minute workload samples. Consequently, most often, when a given workload is reached no failure has occurred (i.e. failures are quite infrequent).

If $z(x)$ increases with x, it should be clear that there is an increasing risk of a failure as the workload variable increases. If, however, $z(x)$ remains constant for increasing x, we may surmise that no increased risk is involved.

Note that in our definition of load hazard we have removed the variability of system load by using the conditional probability $g(x)$. This of course is not true in practice since load is best described as a random variable with a probability distribution; it is simply the associated load distribution, $\ell(x)$, defined above. In order to determine the hazard for a particular load pattern, we must multiply the associated load probability by the hazard calculated in (1). Denoting by $z_a(x)$ the transformed hazard, we have

$$z_a(x) = z(x)\,\ell(x) \tag{2}$$

We refer to the hazard $z(x)$, as defined in (1), as the _fundamental_ hazard. This is because it can be thought of as an inherent property of a particular system and is not subject to varying load patterns. When a varying load pattern is taken into account, it can be thought of as "picking out" aspects of the fundamental hazard function. This hazard $z_a(x)$ defined in (2) will be referred to as the _apparent_ hazard, since it is closely dependent on the load distribution.

6. HAZARD PLOTS

The generation of the hazard plots and associated statistics involved extensive data processing. In each hazard plot, $z(x)$ or $z_a(x)$ is calculated and plotted as a function of a chosen workload variable, x. In developing hazard plots for the load-failure data, there is an important difference between the real and the artificially created data. This lies in the fact that, while an artifical data base has specific dependencies seeded into it, in the real world, failures can occur due to a number of causes. Examples are: temperature, humidity, random noise, mechanical failures, and design errors, some of which are unrelated to our study. Those factors not related to load can be expected to behave as noise in a load-failure analysis. If these other factors are predominant, we can expect to find no discernable pattern in our hazard plots i.e. they should appear as uncorrelated clouds. This is well understood in any statistical study of dependencies.

An easily discernable pattern, on the other hand, would indicate that the load-failure dependency dominates others. The strength of such a relationship can be measured through regression. Figures 10, 11, and 12 depict the hazard plots for the three selected load parameters. The regression coefficient R^2, which is an effective measure of the goodness of fit, is provided for each plot. Quite simply, it measures the amount of variability in the data that can be accounted for by the regression model. R^2 values of greater than 0.6 (corresponding to an $R > 0.75$) are generally interpreted as strong relationships[7] [Younger 79]. It can be seen that the hazards are increasing with each of the load parameters shown. The relationship is particularly strong with SYSCPU and IO, although other measures such as EXCP, SVC and PROG (plots not shown), also correlate strongly. Note that these measures in one way or another measure the interactive workload with some degree of overlap and have different degrees of variability. TOTCPU, a general measure of execution also correlates moderately strongly. In addition, it is seen that the workload-failure relationship is highly non-linear. This appears to indicate that toward the existence of a threshold beyond which the system worsens very rapidly.

It is interesting to note that most of the estimated hardware failures were failures in main memory. An analysis of these failures by time of day showed that they generally occur during the period when the main memory access rate and the interactive workload measures (e.g SYSCPU and IO) are the highest (i.e. during prime time).[8]

---

[7] The range of $|R|$ from 0 to 1 is typically divided as follows: (0, 0.25) moderately weak; (0.25, 0.5) moderate; (0.5, 0.75) moderately strong; (0.75, 1.0) strong.

[8] We also note that, at SLAC, most the heavy compute bound jobs, which require relatively low overhead are run at night, when the paging rate and consequently the main memory access rate is relatively low.

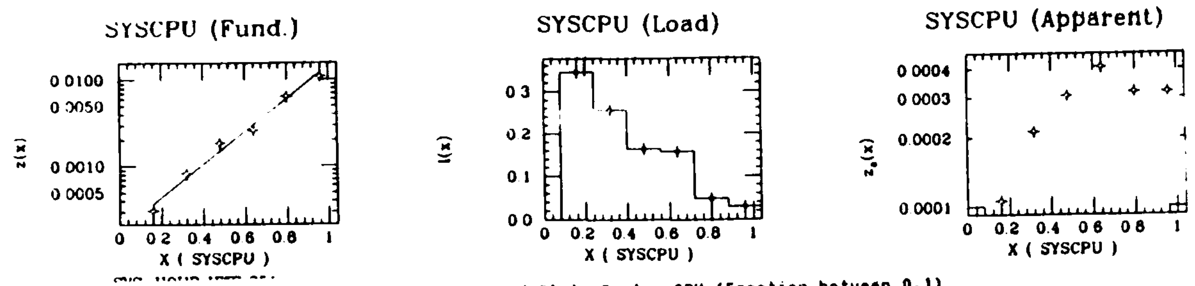


Figure 10: Hazard Plot: System CPU (Fraction between 0,1)
The vertical scale is exponential in these plots, indicating that the hazard is rising sharply at peak loads.

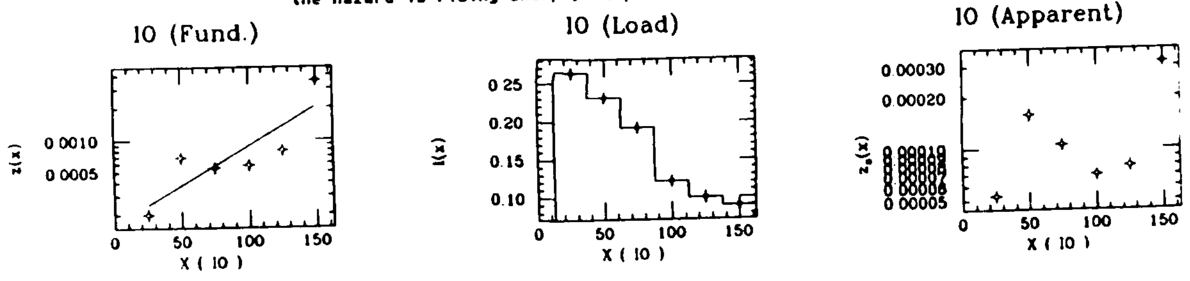


Figure 11: Hazard Plot: IO (Per second)

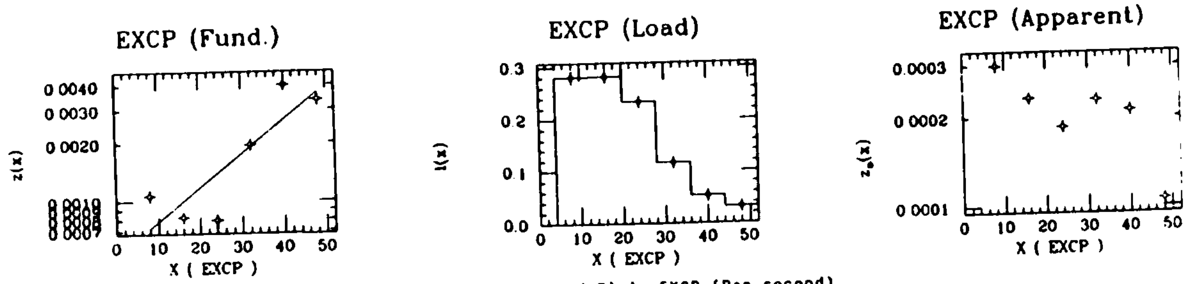


Figure 12: Hazard Plot: EXCP (Per second)

7. DISCUSSION AND CONCLUDING REMARKS

The analysis shows that there is a strong load dependency of internal CPU errors at SLAC. The observed tendency is present in three years of load data analyzed. This is significant because our previously reported results could only provide us with an external view of permanent system and component failures. By examining the CPU error generation process we have been able to study the inner behavior of the system and its reaction to errors. Consequently, we have gathered the best data possible. A load-failure relationship found at this level must, in our view, be a fundamental phenomenon.

The analysis procedure has been demonstrated on artifically created data base seeded with failures. The two hazard models proposed clearly differentiate between <u>fundamental</u> (or inherent) and <u>apparent</u> load dependent failures. An estimate of the fundamental hazard $z(x)$, provides the basic load-failure relationship. The apparent hazard $z_a(x)$ estimates how $z(x)$ is modified by the load probabilities. It is, in principle, possible that even when no inherent relationship exists between load and failures, we could conceivably obtain an apparent dependency simply due to the fact that some load values occur more frequently than others. Alternatively, we can have the reverse situation where an increasing fundamental hazard is transformed into a non-increasing or even decreasing apparent hazard by a distinctive load distribution.

As with any statistical analysis, this is not proof in itself. More measurements and experiments are necessary to further study this problem. However, the increasing body of evidence accumulated on different computers with differing load and failure patterns shows that workload should be considered as a factor relating to reliability. Workload can be thought of as a stress on the system, with greater stresses resulting in greater risk of failure. In view of our previous results, we believe that the error process which ensues is composed of two separate effects. The first is the (constant) inherent failure rate. This is determined through classical reliability techniques [Shooman 68], taking into consideration such factors as topology, redundancy etc. The second is the utilization-induced failure rate. This rate is dependent upon both the absolute level of system utilization and the rate of change of that level. By an absolute level we mean an obviously measurable level; e.g., CPU utilization, memory occupancy, etc. Through the rate of change of utilization we are attempting to measure the rate at which transitions occur between various system states, e.g. the transitions of the CPU into and out of the busy state. In most cases the effect of this stress is not permanent, since most errors are transient [Iyer 82b]. However, as demonstrated in this paper, there is a significant contribution due to hard device failures in the CPU and main storage.

A preliminary examination of the semiconductor device literature shows that some experimental and quantitative evidence exists to support support our results. For example the effect of transient and intermittent loading on the rating of power devices has been studied at length (see [Ivalo 61] and [Blackburn 74] for details). It is well known that the duty cycle of the pulse is an important parameter in the rating of these devices for pulsed operation. [Owen 80] describes practical methods commonly employed to evaluate the thermal effects of repetitive pulsed loading. Detailed analytical and experimental analysis of both steady state and transient thermal behaviour is discussed in [Newell 75].

There is also evidence in the general reliability literature which relates low and high usage rates, of avionic and navigational equipment with corresponding reliability behaviour (see [Shurman 78] and [Kujowski 78] for details). It is to be noted that in each of these two studies a significant component of the system was electronic or digital. Our measurements show that the effect is not negligible in smaller devices. The design of computer systems will be greatly aided if this type of analysis can help uncover cause and effect relationships in hardware errors.

8. ACKNOWLEDGMENTS

We gratefully acknowledge Prof. E.J. McCluskey for his continued interest in this work and for extensive discussions throughout the period of this study. We would also like to thank the SLAC computional facility, in particular, Ted Johnston for providing the computing resources and access to the data. In addition, we thank Dr. Y. Min, M. Cortes and H. Amer for their careful reading of an earliar draft of this paper.

This work was supported in part by the Department of the Army under Contract Number DAAG29-82-K-0105 and by the Department of Energy under Contract Number DE-AC03-76F00515.

REFERENCES

[Arsenault 80] J. E. Arsenault and J. A. Roberts, Reliability and Maintainability of Electronic Systems, Computer Science Press, Potomic, Maryland, 1980.

[Blackburn 74] D. L. Blackburn and F. F. Oettinger, "Transient thermal response of power transistors," in IEEE PESC Conf. Rec., pp. 140-148, June 1974.

[Butner 80] S. E. Butner and R. K. Iyer "A Statistical Study of Reliability and System Load at SLAC", Digest, Tenth International Symposium on Fault Tolerant Computing, Kyoto, Japan, October 1980.

[Castillo 80] X. Castillo and D. P. Siewiorek, "A Performance Reliability Model for Computing Systems", Digest, Tenth International Symposium on Fault Tolerant Computing, Kyoto, Japan, October 1980.

[Castillo 81] X. Castillo and D. P. Siewiorek, "Workload, Performance and Reliability of Digital Computing Systems," Digest, Eleventh International Symposium on Fault-Tolerant Computing, Portland, Maine, June 1981, pp. 84-89.

[Castillo 82] X. Castillo and D. P. Siewiorek, " A Workload Dependent Software Reliability Prediction Model," Digest, Twelvetn International Symposium on Fault-Tolerant Computing, Santa Monica, California, June 1982.

[Feller 68] W. Feller, An Introduction to Probability Theory and its Applications, John Wiley and Sons, 1968.

[Gunther 80] N. L. Gunther and W. C. Carter, "Remarks on the Probability of Detecting Faults", Digest, Tenth International Symposium on Fault Tolerant Computing, Kyoto, Japan, October 1980.

[IBM 73] IBM Corp., OS/VS System Management Facilities (SMF), Order No. GC35-0004, 1973.

[IBM 79] IBM Corp., OS/VS, DOS/VSE, VM/370 Environmental Recording Editing and Printing (EREP) Program, Order No. GC28-0772, 1979.

[IBM 81] IBM Corp., IBM System/370 Principles of Operation, Order No. GA22-7000-8, 1981.

[Ivalo 62] V. E. S. Ivalo, "Pulse Rating Charts for the Loadability of Semiconductor Devices", Electronic Applications, vol. 22, No 4, pp. 148-162, 1962.

[Iyer 82a] R. K. Iyer, S. E. Butner, and E. J. McCluskey, "A Statistical Failure/Load Relationship; Results of a Multi-Computer Study," IEEE Transactions on Computers, July 1982.

[Iyer 82b] R. K. Iyer and D. J. Rossetti, "A Statistical Load Dependency of CPU Errors at SLAC", Digest, Twelveth International Symposium on Fault Tolerant Computing, Santa Monica, California, June 1982.

[Kujowski 78] G. F. Kujowski and E. A. Rypka, "Effects of On-Off Cycling on Equipment Reliability," 1978 Reliability and Maintainability Symposium, pp. 225-230, 1978.

[Newell 75] W. E. Newell, "Transient thermal analysis of solid state power devices - Making a dreaded process easy," IEEE PESC Conf. Rec. , June 1975.

[Owen 80] G. Owen, "Thermal management techniques keep semiconductors cool," Electronics, pp. 135-142, September 1980.

[Rossetti 81] D. J. Rossetti and R. K. Iyer, "A Software System for Reliability and Workload Analysis", CRC Tech. Rpt 81-18, Center for Reliable Computing, Computer Systems Laboratory, Stanford Univ., Stanford, California, December 1981.

[Rossetti 82] D. J. Rossetti and R. K. Iyer, "Software Related Failures on IBM 3081: A Relationship with System Utilization", Proc. COMPSAC 82, Chicago, Illinois, November 1982.

[SAS 79] SAS Institute Inc., SAS User's Guide, 1979 Edition, 1979.

[Shooman 68] M. L. Shooman, Probabilistic Reliability: An Engineering Approach, McGraw Hill, 1968.

[Shurman 78] M. B. Shurman, "Time Dependent Failures Rates for Jet Aircraft", 1978 Reliability and Maintainability Symposium, pp. 198-201, 1978.

[Younger 79] M. S. Younger, A Handbook for Linear Regression, Wadsworth Inc., 1979.